Short Course on Numerical Tools on

The ACTS Collection:

Robust and High Performance Libraries for Computational Sciences

SIAM CSE-05 - Orlando, Florida
February 11, 2005

Tony Drummond and Osni Marques
Lawrence Berkeley National Laboratory
acts-support@nersc.gov
{ladrummond, oamarques}@lbl.gov

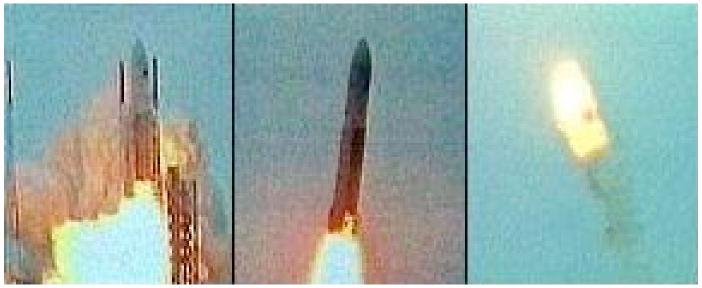
Jose E. Roman
Polytechnic University of Valencia
jroman@dsic.upv.es

Outline

OUTLINE:	
• Introduction to ACTS (Tony)	8:30 AM
• ScaLAPACK (Osni)	8:50 AM
• SuperLU (Osni)	9:10 AM
• Introduction to SLEPc (Jose) ○ Brief introduction to PETSc ○ SLEPc	9:30 AM
BREAK (20 Mins)	10:40 AM
• Optimization Tools (Tony) ○ TAO ○ OPT++	11:00 AM
• PyACTS (Tony)	11:20 AM
• Code Profiling for Performance ○ Libraries Automatic tuning (Tony) ○ TAU (Osni)	11:40 AM
• Future Plans (Osni)	12:20 PM

Motivation

On February 25, 1991, during the Gulf War, an American Patriot Missile battery in Dharan, Saudi Arabia, failed to track and intercept an incoming Iraqi Scud missile. The Scud struck an American Army barracks, killing 28 soldiers and injuring around 100 other people. The problem was an inaccurate calculation of the time since boot due to computer arithmetic errors.



On June 4, 1996, an Ariane 5 rocket launched by the European Space Agency exploded just forty seconds after its lift-off from Kourou, French Guiana. The rocket was on its first voyage, after a decade of development costing $7 billion. The problem was a software error in the inertial reference system. Specifically a 64 bit floating point number relating to the horizontal velocity of the rocket with respect to the platform was converted to a 16 bit signed integer.

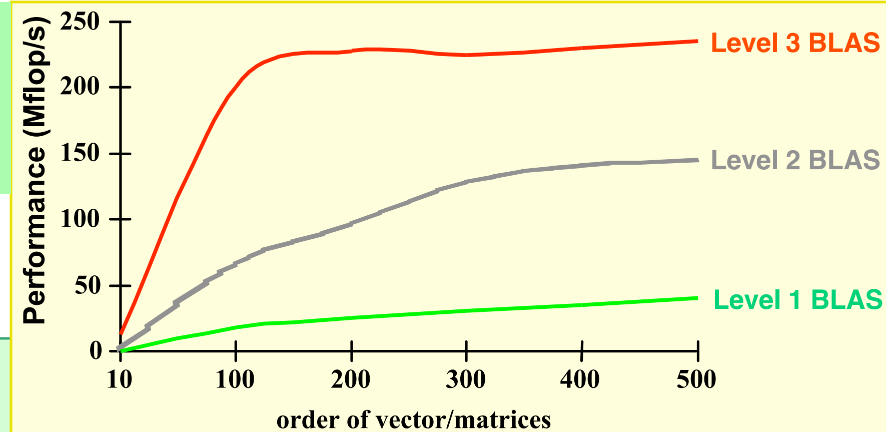
On August 23,1991, the first concrete base structure for the Sleipner A platform sprang a leak and sank under a controlled ballasting operation during preparation for deck mating in Gandsfjorden outside Stavanger, Norway. The post accident investigation traced the error to inaccurate finite element approximation of the linear elastic model of the tricell (using the popular finite element program NASTRAN). The shear stresses were underestimated by 47% leading to insufficient design. In particular, certain concrete walls were not thick enough.

http://wwwzenger.informatik.tu-muenchen.de/persons/huckle/bugse.html

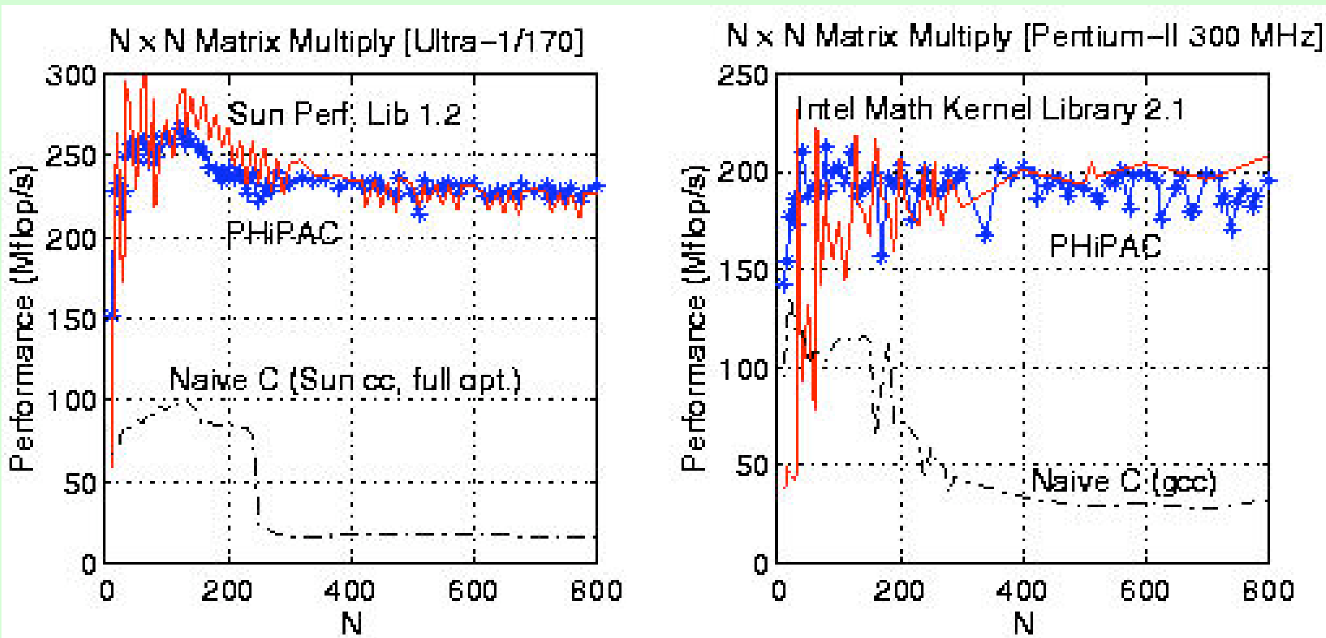
Motivation

Automatic Tuning

- PHiPAC: *www.icsi.berkeley.edu/~bilmes/phipac*
- ATLAS: *www.netlib.org/atlas*

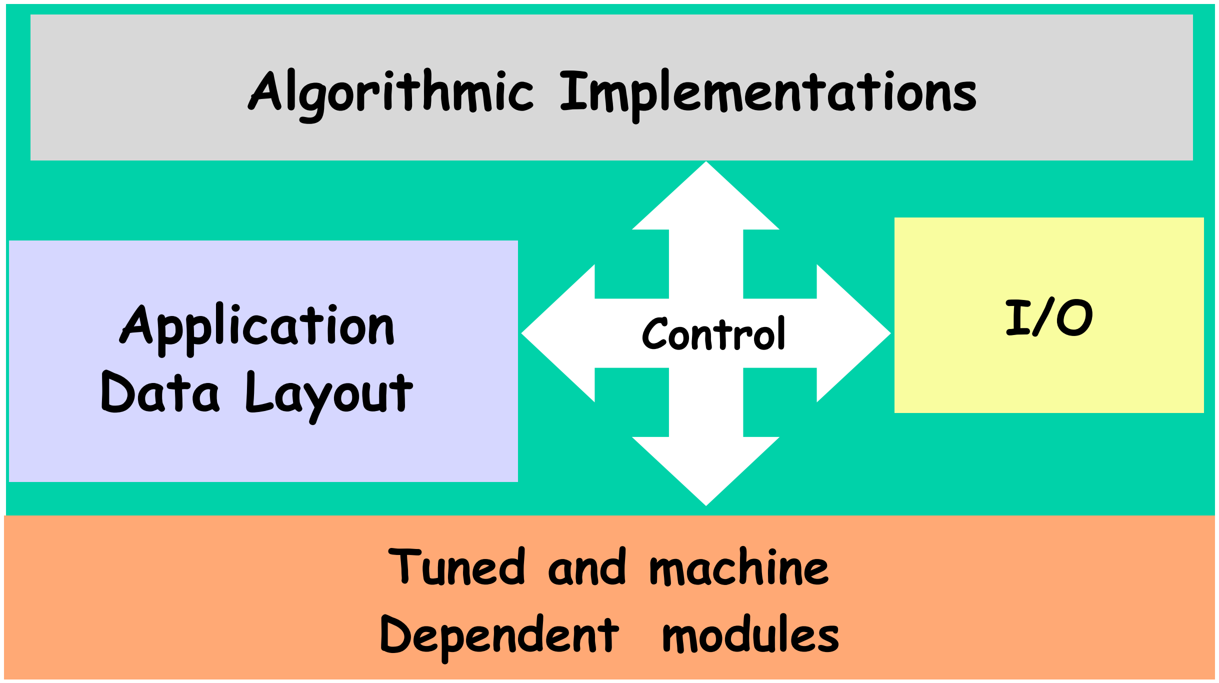


PHiPAC: C = A * B



Motivation- Why do we need software libraries?

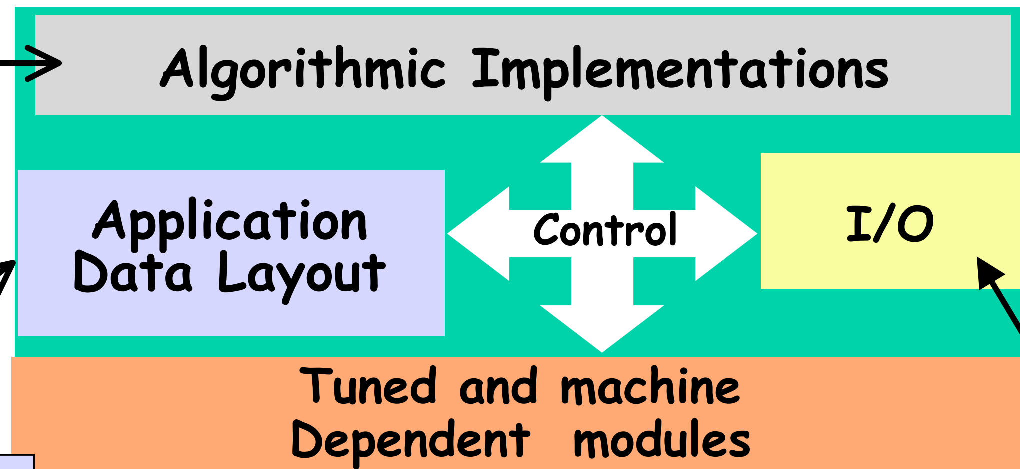
Large Scientific Codes:
A Common Programming Practice



Motivation- Why do we need software libraries?

New Architecture:
- May or may not need re-rewriting

New Developments:
- Difficult to compare



New Architecture:
- Extensive re-rewriting

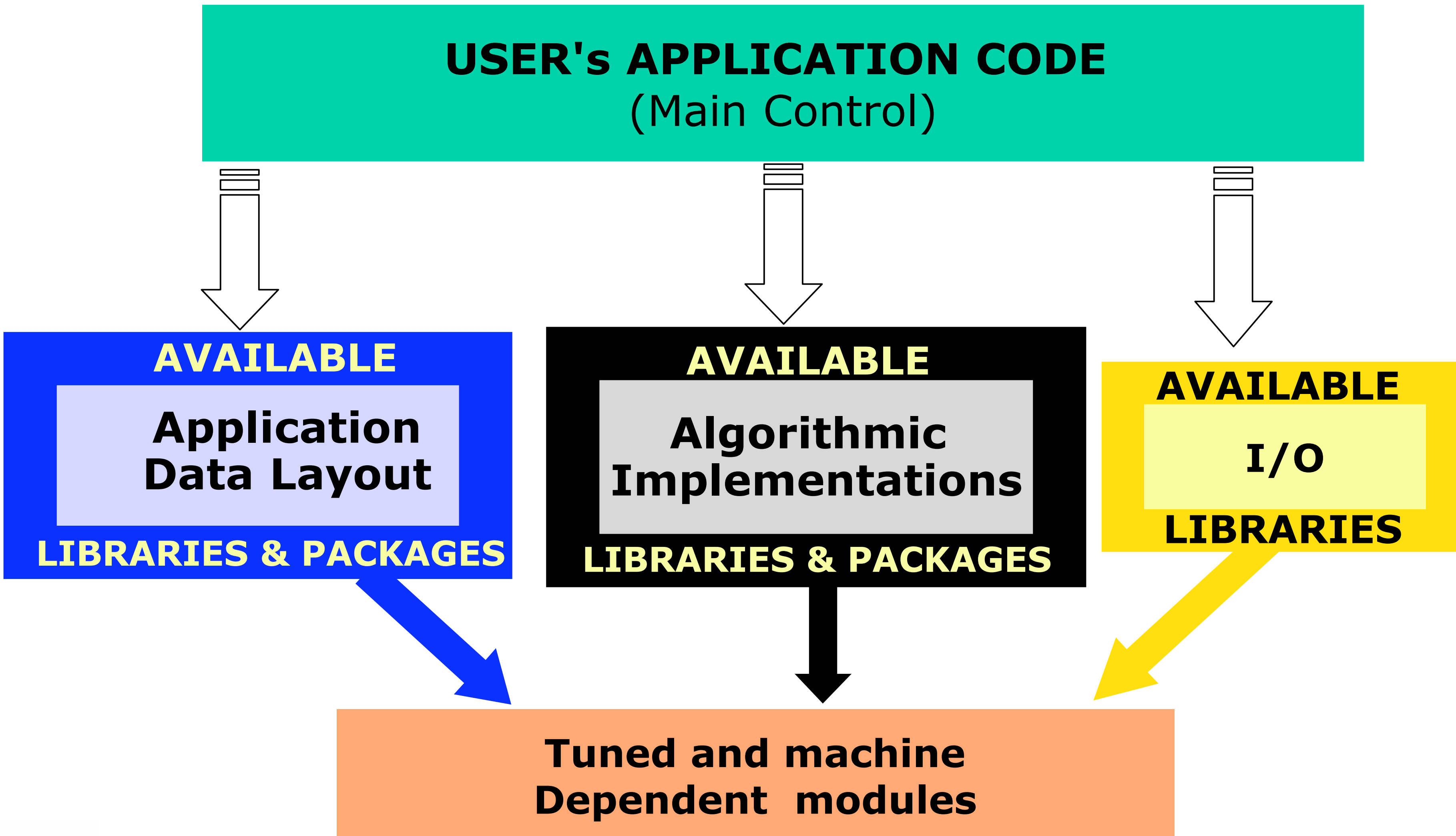
New or extended Physics:
- Extensive re-rewriting or increase overhead

New Architecture:
- Minimal to Extensive rewriting

New Architecture or S/W
- Extensive tuning
- May require new programming paradigms
- **Difficult to maintained!**

Motivation- Why do we need software libraries?

An Alternative Approach



Good Reasons to Use Libraries

- **Productivity**
  - Time to the first solution (prototype)
  - Time to solution (production)
  - Long term and upgrades
- **Complexity**
  - Increasingly sophisticated models
  - Model coupling
  - Interdisciplinary Nature
- **Performance**
  - Increasingly complex algorithms
  - Increasingly complex architectures
  - Increasingly demanding applications

The ACTS Collection?

http://acts.nersc.gov

- Advanced CompuTational Software Collection
- Tools for developing parallel applications
- ACTS started as an “umbrella” project

Goals

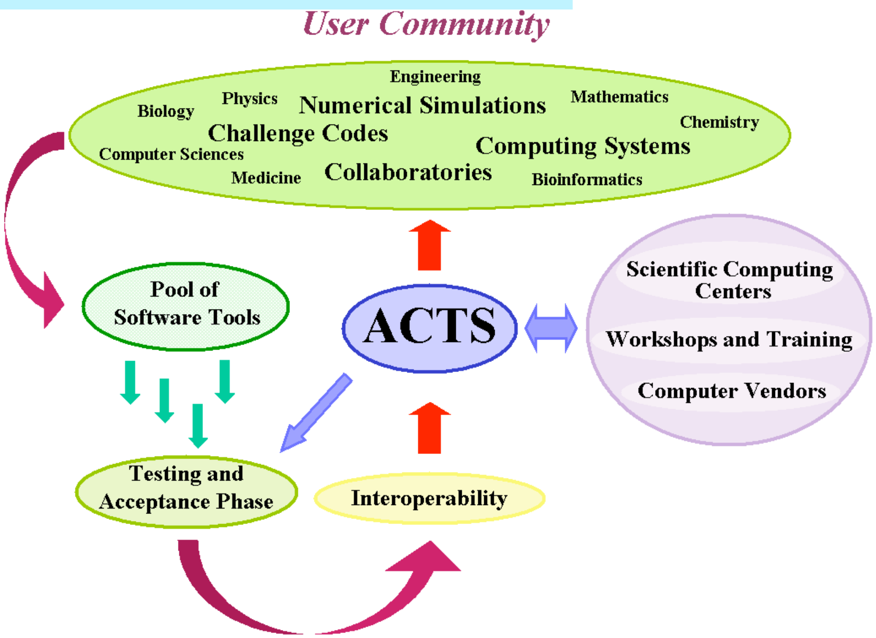
- *Extended support for experimental software*
- *Make ACTS tools available on DOE computers*
- *Provide technical support (acts-support@nersc.gov)*
- *Maintain ACTS information center (http://acts.nersc.gov)*
- *Coordinate efforts with other supercomputing centers*
- *Enable large scale scientific applications*
- *Educate and train*

- ***High***
  - Intermediate level
  - Tool expertise
  - Conduct tutorials
- ***Intermediate***
  - Basic level
  - Higher level of support to users of the tool
- ***Basic***
  - Help with installation
  - Basic knowledge of the tools
  - Compilation of user’s reports

Why is ACTS unique?

- Provides pointers and documentation about software tools.
- Accumulates the expertise and user feedback on the use of the software tools and scientific applications that used them:
  - independent software evaluations
  - participation in the developer user groups e-mail list
  - leverage between tool developers and tool users
  - presentation of a gallery of applications
  - workshops and tutorials
  - tool classification
  - support

4th ACTS Workshop, August 5-8, 2004, Berkeley, CA



ACTS Tools

Category	Tool	Functionalities
Numerical $Ax = b$ $Az = \lambda z$ $A = U\Sigma V^T$ PDEs ODEs M	**Aztec**	Algorithms for the iterative solution of large sparse linear systems.
	Hypre	Algorithms for the iterative solution of large sparse linear systems, intuitive grid-centric interfaces, and dynamic configuration of parameters.
	PETSc	Tools for the solution of PDEs that require solving large-scale, sparse linear and nonlinear systems of equations.
	OPT++	Object-oriented nonlinear optimization package.
	SUNDIALS	Solvers for the solution of systems of ordinary differential equations, nonlinear algebraic equations, and differential-algebraic equations.
	ScaLAPACK	Library of high performance dense linear algebra routines for distributed-memory message-passing.
	SuperLU	General-purpose library for the direct solution of large, sparse, nonsymmetric systems of linear equations.
	TAO	Large-scale optimization software, including nonlinear least squares, unconstrained minimization, bound constrained optimization, and general nonlinear optimization.
Code Development	**Global Arrays**	Library for writing parallel programs that use large arrays distributed across processing nodes and that offers a shared-memory view of distributed arrays.
	Overture	Object-Oriented tools for solving computational fluid dynamics and combustion problems in complex geometries.
Code Execution	**CUMULVS**	Framework that enables programmers to incorporate fault-tolerance, interactive visualization and computational steering into existing parallel programs
	Globus	Services for the creation of computational Grids and tools with which applications can be developed to access the Grid.
	PAWS	Framework for coupling parallel applications within a component-like model.
	SILOON	Tools and run-time support for building easy-to-use external interfaces to existing numerical codes.
	TAU	Set of tools for analyzing the performance of C, C++, Fortran and Java programs.
Library Development	**ATLAS and PHiPAC**	Tools for the automatic generation of optimized numerical software for modern computer architectures and compilers.
	PETE	Extensible implementation of the expression template technique (C++ technique for passing expressions as function arguments).

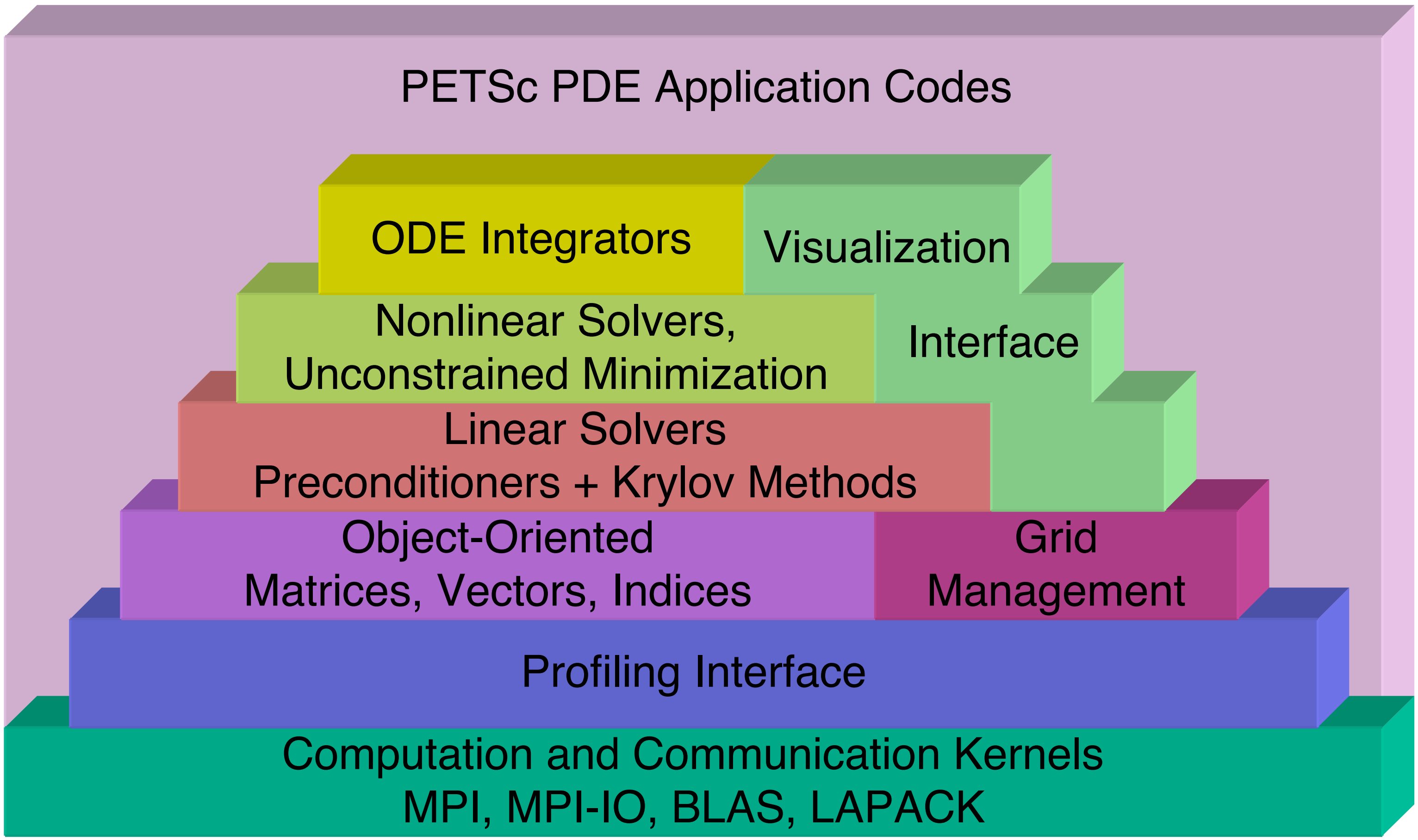
ACTS Numerical Tools: *Functionality*

Computational Problem	Methodology	Algorithms	Library
Systems of Linear Equations	Direct Methods	LU Factorization	ScaLAPACK(dense) SuperLU (sparse)
		Cholesky Factorization	ScaLAPACK
		LDL^T (Tridiagonal matrices)	ScaLAPACK
		QR Factorization	ScaLAPACK
		QR with column pivoting	ScaLAPACK
		LQ factorization	ScaLAPACK

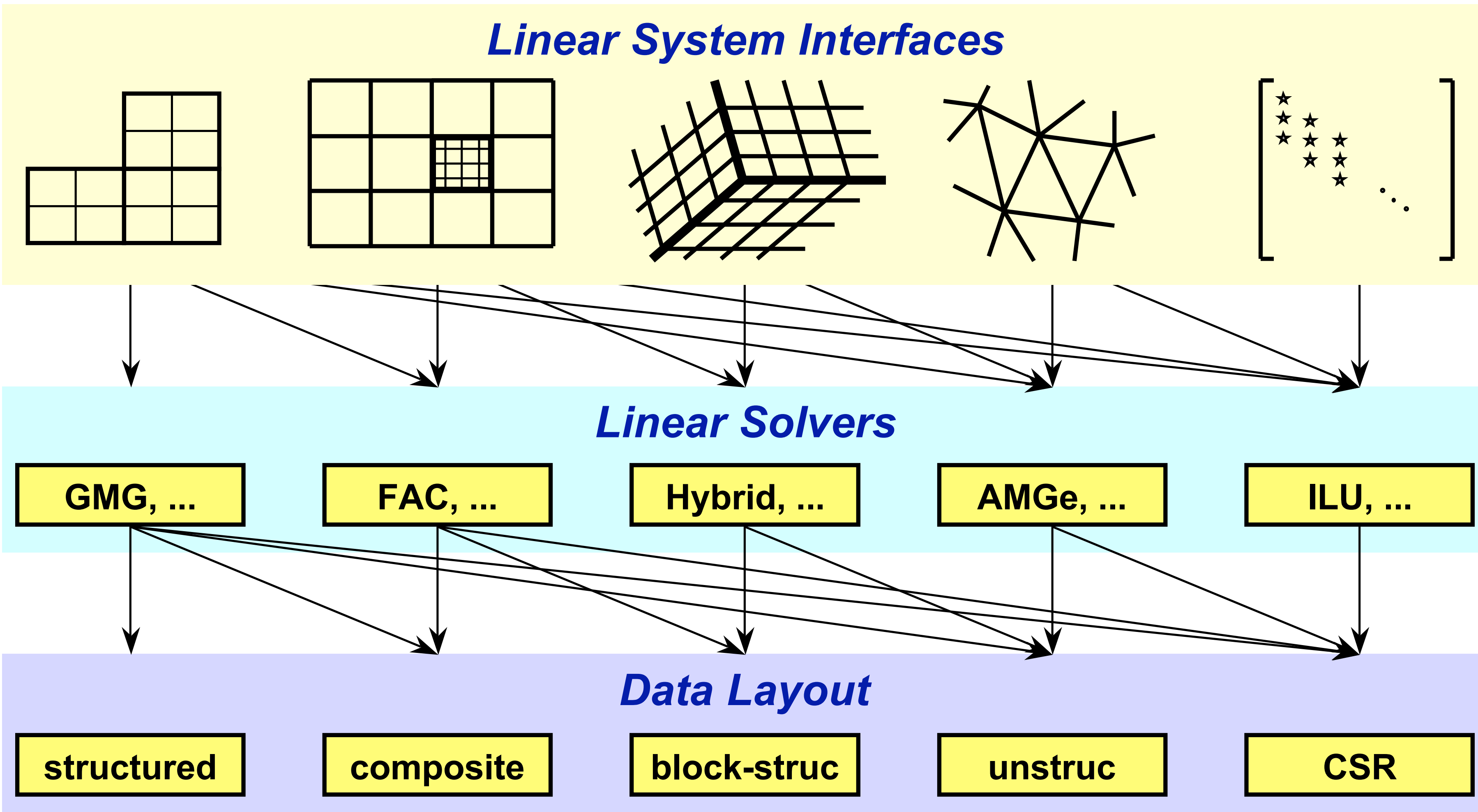
ACTS Numerical Tools: *Functionality*

Computational Problem	Methodology	Algorithms	Library
Systems of Linear Equations (*cont*..)	Iterative Methods	Conjugate Gradient	AztecOO (Trilinos) PETSc
		GMRES	AztecOO PETSc Hypre
		CG Squared	AztecOO PETSc
		Bi-CG Stab	AztecOO PETSc
		Quasi-Minimal Residual (QMR)	AztecOO
		Transpose Free QMR	AztecOO PETSc

Structure of PETSc



Hypre Conceptual Interfaces



INTERFACE TO SOLVERS

List of Solvers and Preconditioners per Conceptual Interface

Solvers	System Interfaces			
	Struct	SStruct	FEI	IJ
Jacobi	X			
SMG	X			
PFMG	X			
BoomerAMG	X	X	X	X
ParaSails	X	X	X	X
PILUT	X	X	X	X
Euclid	X	X	X	X
PCG	X	X	X	X
GMRES	X	X	X	X

ACTS Numerical Tools: *Functionality*

Computational Problem	Methodology	Algorithms	Library
Systems of Linear Equations (*cont*..)	Iterative Methods (*cont*..)	SYMMLQ	PETSc
		Precondition CG	AztecOO PETSc Hypre
		Richardson	PETSc
		Block Jacobi Preconditioner	AztecOO PETSc Hypre
		Point Jocobi Preconditioner	AztecOO
		Least Squares Polynomials	PETSc

ACTS Numerical Tools: *Functionality*

Computational Problem	Methodology	Algorithms	Library
Systems of Linear Equations (*cont*..)	Iterative Methods (*cont*..)	SOR Preconditioning	PETSc
		Overlapping Additive Schwartz	PETSc
		Approximate Inverse	Hypre
		Sparse LU preconditioner	AztecOO PETSc Hypre
		Incomplete LU (ILU) preconditioner	AztecOO
		Least Squares Polynomials	PETSc
	MultiGrid (MG) Methods	MG Preconditioner	PETSc Hypre
		Algebraic MG	Hypre
		Semi-coarsening	Hypre

ACTS Numerical Tools: *Functionality*

Computational Problem	Methodology	Algorithm	Library
Linear Least Squares Problems	Least Squares	$min_x \| b - Ax \|_2$	ScaLAPACK
	Minimum Norm Solution	$min_x \| x \|_2$	ScaLAPACK
	Minimum Norm Least Squares	$min_x \| b - Ax \|_2$ $min_x \| x \|_2$	ScaLAPACK
Standard Eigenvalue Problem	Symmetric Eigenvalue Problem	$Az = \lambda z$ For $A=A^H$ or $A=A^T$	ScaLAPACK (dense) SLEPc (sparse)
Singular Value Problem	Singular Value Decomposition	$A = U\Sigma V^T$ $A = U\Sigma V^H$	ScaLAPACK (dense) SLEPc (sparse)
Generalized Symmetric Definite Eigenproblem	Eigenproblem	$Az = \lambda Bz$ $ABz = \lambda z$ $BAz = \lambda z$	ScaLAPACK (dense) SLEPc (sparse)

ACTS Numerical Tools: *Functionality*

Computational Problem	Methodology	Algorithm	Library
Non-Linear Equations	Newton Based	Line Search	PETSc
		Trust Regions	PETSc
		Pseudo-Transient Continuation	PETSc
		Matrix Free	PETSc

ACTS Numerical Tools: *Functionality*

Computational Problem	Methodology	Algorithm	Library
Non-Linear Optimization	Newton Based	Newton	OPT++ TAO
		Finite-Difference Newton	OPT++ TAO
		Quasi-Newton	OPT++ TAO
		Non-linear Interior Point	OPT++ TAO
	CG	Standard Non-linear CG	OPT++ TAO
		Limited Memory BFGS	OPT++
		Gradient Projections	TAO
	Direct Search	No derivate information	OPT++

ACTS Numerical Tools: *Functionality*

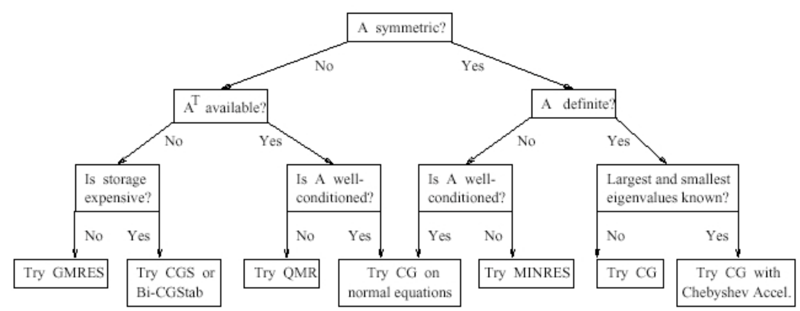
Computational Problem	Methodology	Algorithm	Library
Non-Linear Optimization (cont..)	Semismoothing	Feasible Semismooth	TAO
		Unfeasible semismooth	TAO
Ordinary Differential Equations	Integration	Adam-Moulton (Variable coefficient forms)	CVODE (SUNDIALS) CVODES
	Backward Differential Formula	Direct and Iterative Solvers	CVODE CVODES
Nonlinear Algebraic Equations	Inexact Newton	Line Search	KINSOL (SUNDIALS)
Differential Algebraic Equations	Backward Differential Formula	Direct and Iterative Solvers	IDA (SUNDIALS)

Software Selection

$$\text{Example: } Ax = b$$

- Use a direct solver ($A=LU$) if
  - Time and storage space acceptable
  - Iterative methods don't converge
  - Many b's for same A
- Criteria for choosing a direct solver
  - Symmetric positive definite (SPD)
  - Symmetric
  - Symmetric-pattern
  - Unsymmetric
- Row/column ordering schemes available
  - MMD, AMD, ND, graph partitioning
- Hardware

Build a preconditioning matrix K such that $Kx=b$ is much easier to solve than $Ax=b$ and K is somehow "close" to A (incomplete LU decompositions, sparse approximate inverses, polynomial preconditioners, preconditioning by blocks or domains, element-by-element, etc). See *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods (Demmel et. al.)*



Software Selection

```
CALL BLACS_GET( -1, 0, ICTXT )
 CALL BLACS_GRIDINIT( ICTXT, 'Row-major', NPROW, NPCOL )
:
 CALL BLACS_GRIDINFO( ICTXT, NPROW, NPCOL, MYROW, MYCOL )
:
:
 CALL PDGESV( N, NRHS, A, IA, JA, DESCA, IPIV, B, IB, JB, DESCB,
 $              INFO )
```

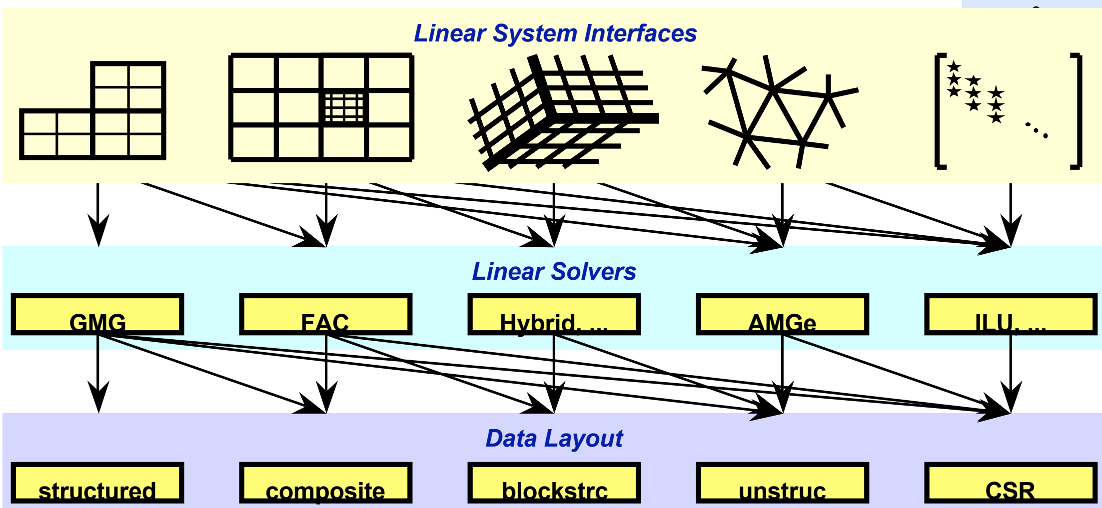
Language Calls

Command lines

- -ksp_type [cg,gmres,bcgs,tfqmr,…]
- -pc_type [lu,ilu,jacobi,sor,asm,…]

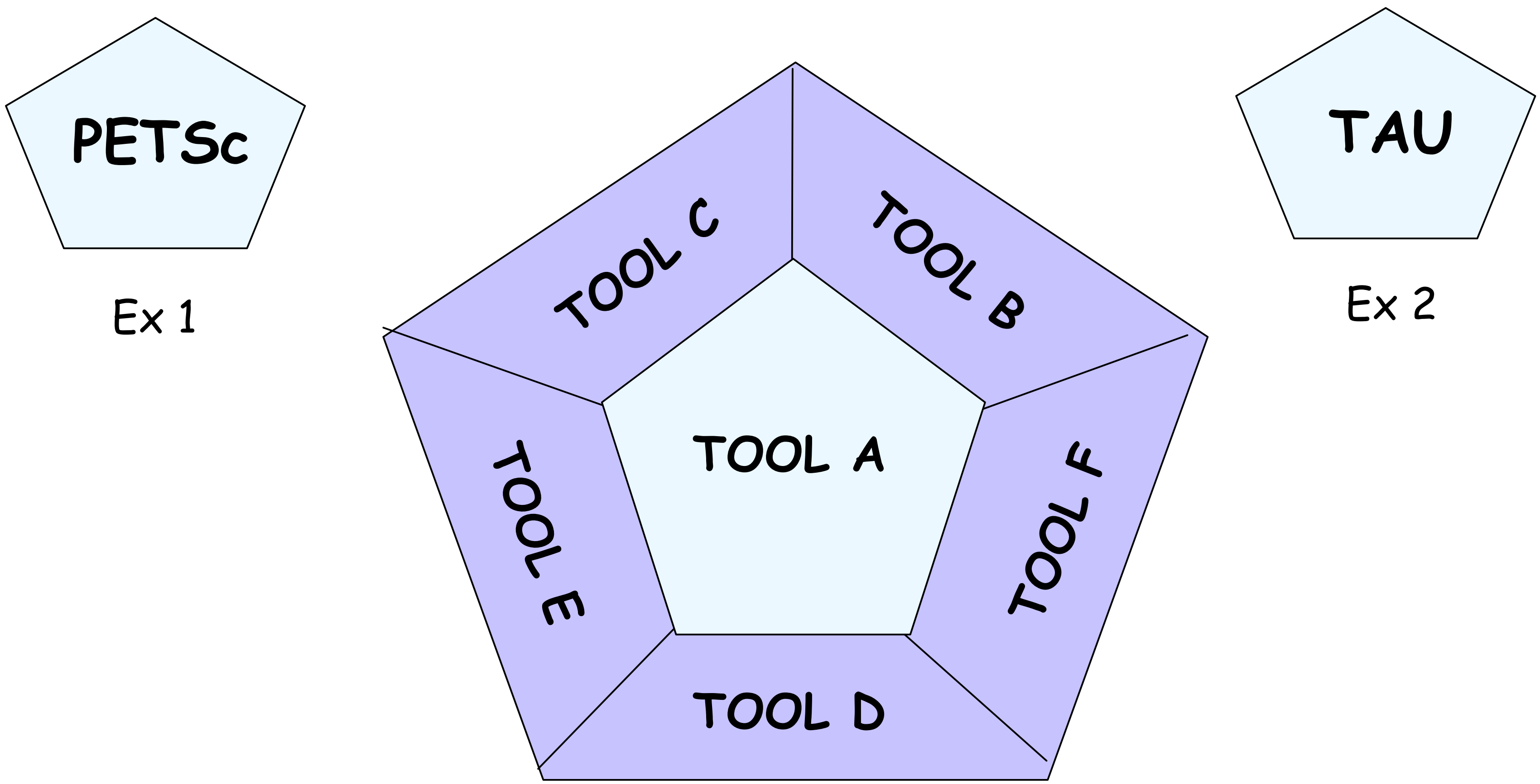
More advanced:

- -ksp_max_it <max_iters>
- -ksp_gmres_restart <restart>
- -pc_asm_overlap <overlap>
- -pc_asm_type [basic,restrict,interpolate,none]

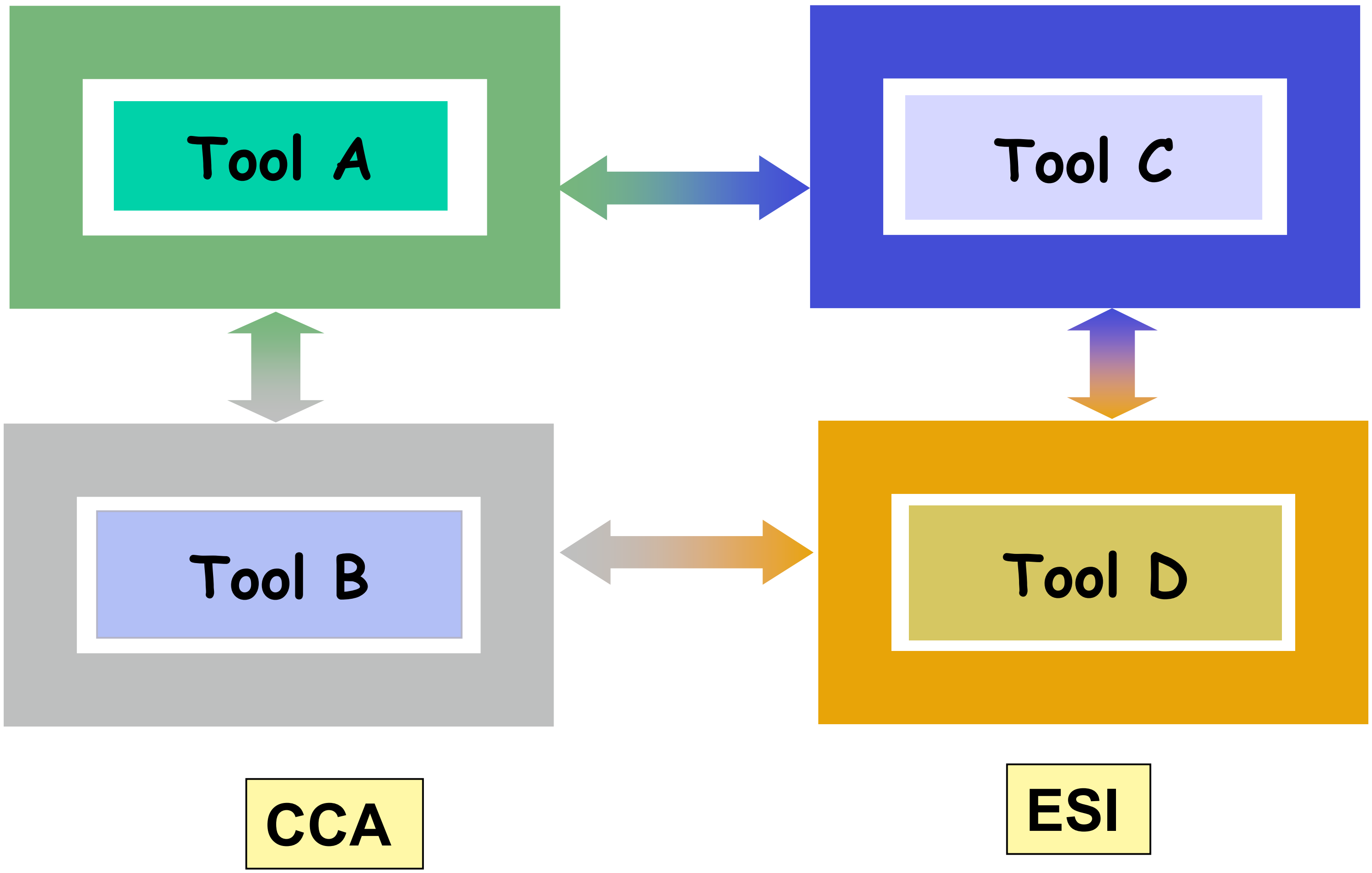


Problem Domain

Tool Interoperability
Tool-to-Tool



Component Technology!



PSE's and Frameworks

PMatlab

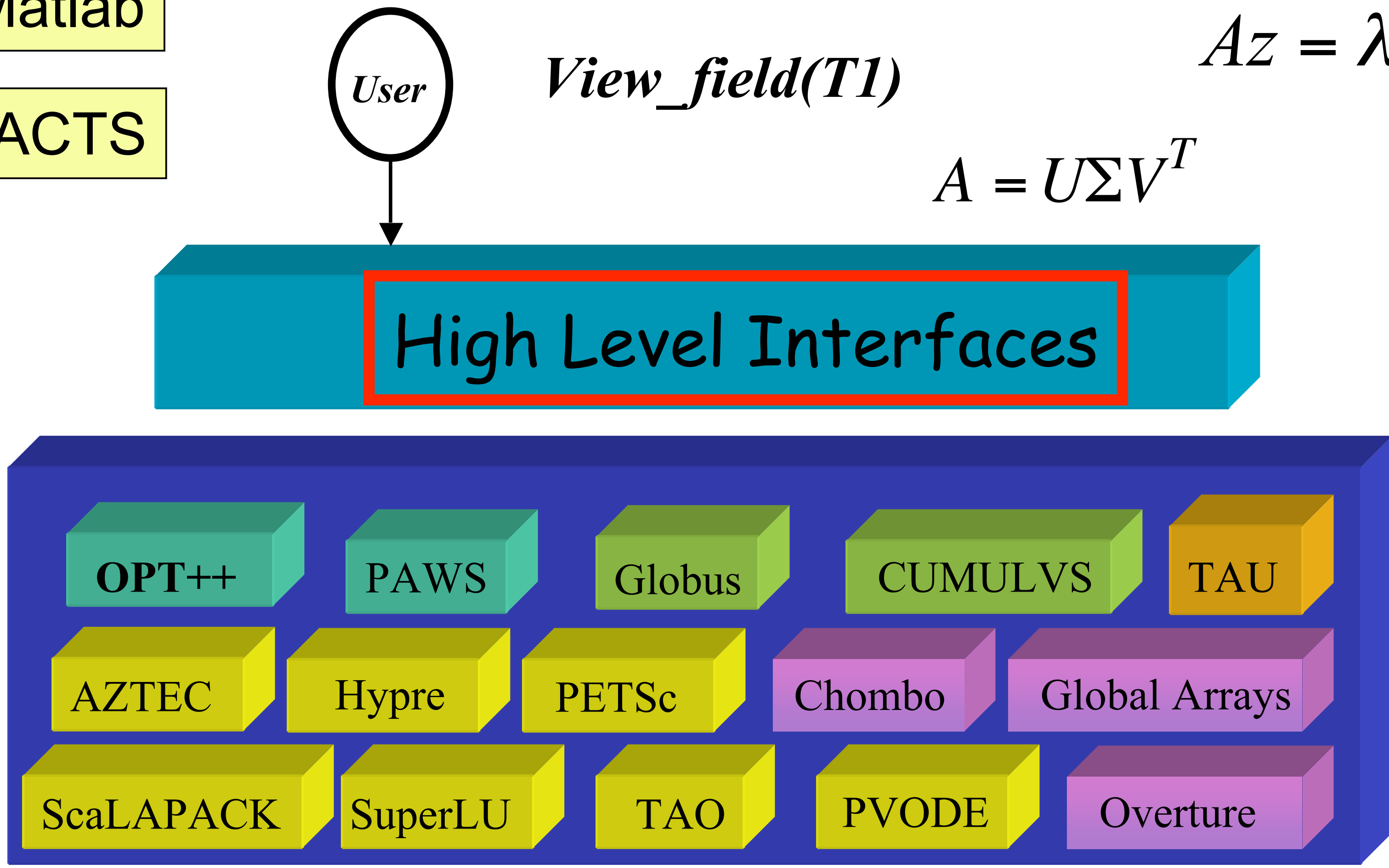
PyACTS

$$Ax = b$$

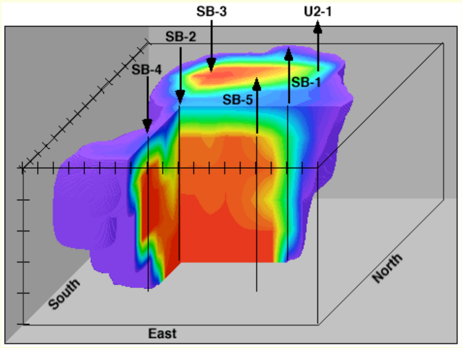
$$Az = \lambda z$$

View_field(T1)

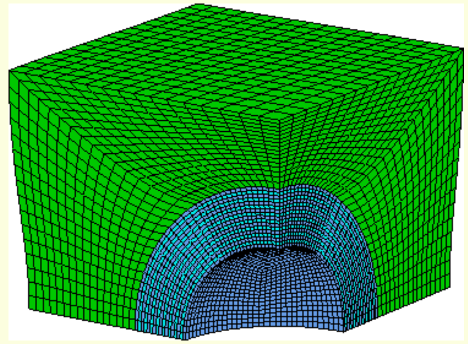
$$A = U\Sigma V^T$$



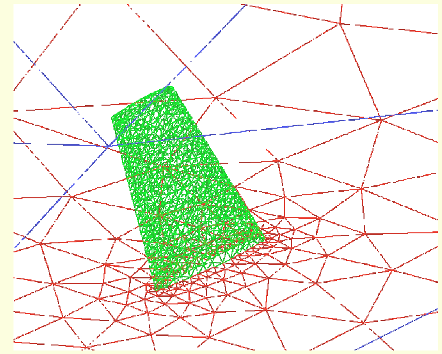
Use of ACTS Tools



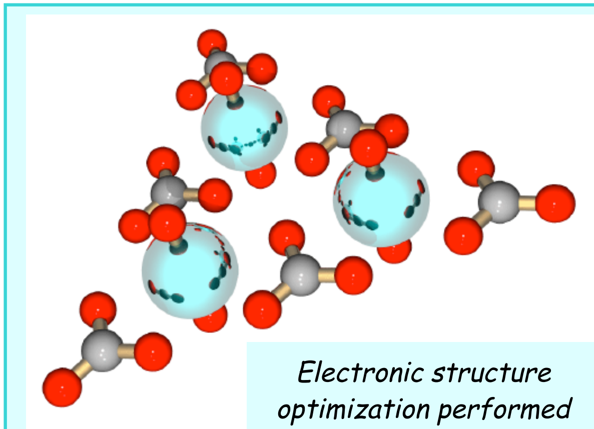
Multiphase flow using PETSc, 4 million cell blocks, 32 million DOF, over 10.6 Gflops on an IBM SP (128 nodes), entire simulation runs in less than 30 minutes (Pope, Gropp, Morgan, Seperhrnoori, Smith and Wheeler).



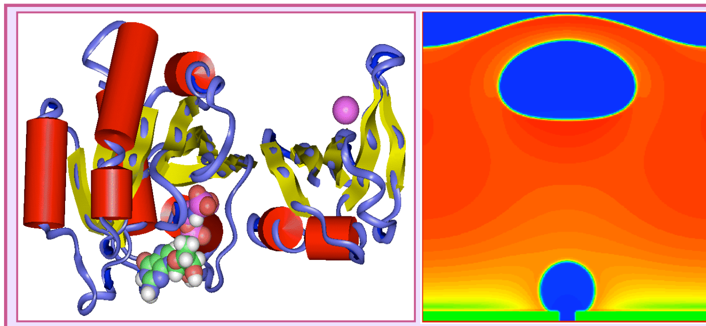
Model of a "hard" sphere included in a "soft" material, 26 million d.o.f. Unstructured meshes in solid mechanics using Prometheus and PETSc (Adams and Demmel).



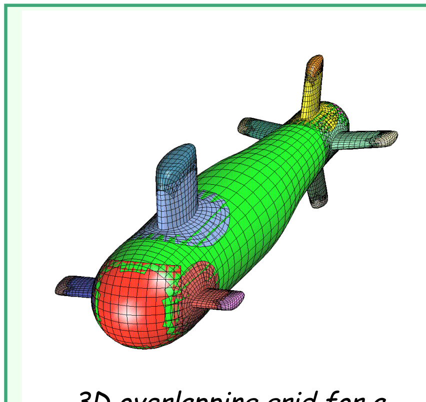
3D incompressible Euler,tetrahedral grid, up to 11 million unknowns, based on a legacy NASA code, FUN3d (W. K. Anderson), fully implicit steady-state, parallelized with PETSc (courtesy of Kaushik and Keyes).



Electronic structure optimization performed with TAO, $(UO_2)_3(CO_3)_6$ (courtesy of deJong).

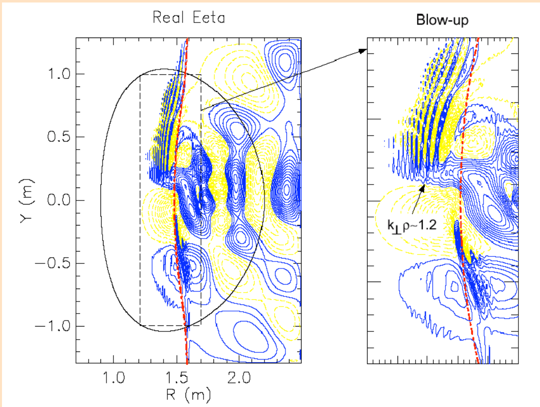


Molecular dynamics and thermal flow simulation using codes based on Global Arrays. GA have been employed in large simulation codes such as NWChem, GAMESS-UK, Columbus, Molpro, Molcas, MWPhys/Grid, etc.

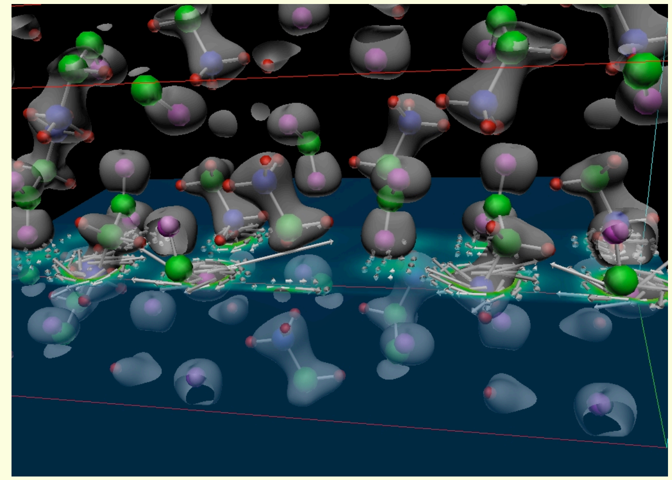


3D overlapping grid for a submarine produced with Overture's module ogen.

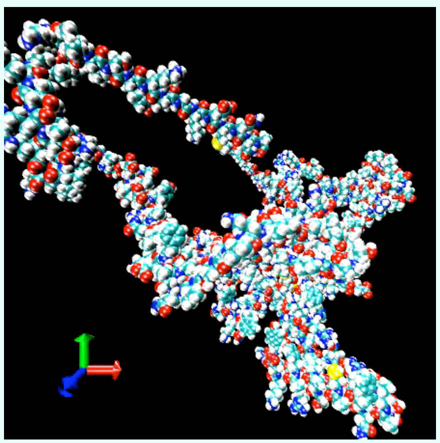
Use of ACTS Tools



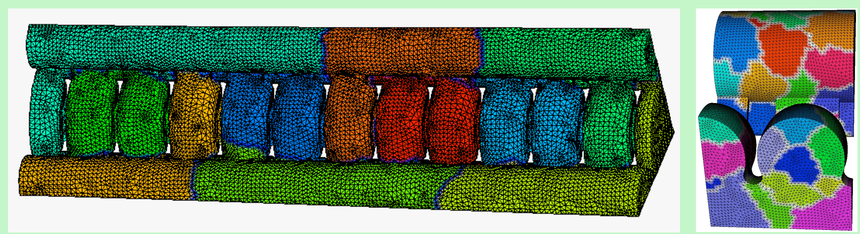
Two ScaLAPACK routines, PZGETRF and PZGETRS, are used for solution of linear systems in the spectral algorithms based AORSA code (Batchelor et al.), which is intended for the study of electromagnetic wave-plasma interactions. The code reaches 68% of peak performance on 1936 processors of an IBM SP.



Induced current (white arrows) and charge density (colored plane and gray surface) in crystallized glycine due to an external field (Louie, Yoon, Pfrommer and Canning), eigenvalue problems solved with **ScaLAPACK**.



OPT++ is used in protein energy minimization problems (shown here is protein T162 from CASP5, courtesy of Meza , Oliva et al.)



Omega3P is a parallel distributed-memory code intended for the modeling and analysis of accelerator cavities, which requires the solution of generalized eigenvalue problems. A parallel exact shift-invert eigensolver based on PARPACK and **SuperLU** has allowed for the solution of a problem of order 7.5 million with 304 million nonzeros. Finding 10 eigenvalues requires about 2.5 hours on 24 processors of an IBM SP.